Adaptive mesh refinement techniques for well-balanced schemes for shallow water flows

Pep Mulet

Joint work with Rosa Donat and Anna Martínez Gavara
Grup ANIMS, Dpt. Matemàtica Aplicada, Univ. València

Benasque, August 2011

Outline

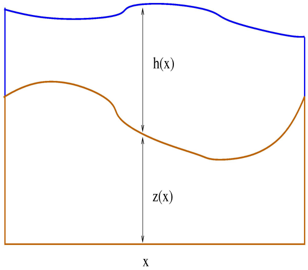
1. Shock capturing schemes for Shallow water flows
2. Adaptive Mesh Refinement
   - Adaptive schemes
   - Grid hierarchy
3. Well-balanced AMR
   - Well-balanced schemes
   - Homogeneous discretization for SWE
   - Well-balanced interpolation
4. Numerical results
   - Numerical results
5. Conclusions

Shallow water flow

- Shallow water equations (SWE) are obtained from incompressible Navier-Stokes equations by depth-averaging and neglecting some terms:

$$h_t + \text{div}(hv) = 0$$

$$(hv)_t + \text{div}(hv \otimes v + \frac{gh^2}{2} I_2) = -gh\nabla z$$



$h \equiv$ water height,
$v = (\bar{v}^x, \bar{v}^y) \equiv$ depth-averaged velocity,
$g \equiv$ gravity acceleration,
$z \equiv$ bottom topography.

- To simplify, we do the exposition in 1D:

$$h_t + (hv)_x = 0$$

$$(hv)_t + (hv^2 + \frac{gh^2}{2})_x = -ghz_x$$

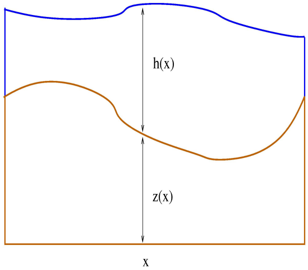
Shallow water flow

- Shallow water equations (SWE) are obtained from incompressible Navier-Stokes equations by depth-averaging and neglecting some terms:

$$h_t + \mathrm{div}(hv) = 0$$

$$(hv)_t + \mathrm{div}(hv \otimes v + \frac{gh^2}{2} I_2) = -gh\nabla z$$

$h \equiv$ water height,
$v = (\bar{v}^x, \bar{v}^y) \equiv$ depth-averaged velocity,
$g \equiv$ gravity acceleration,
$z \equiv$ bottom topography.



- To simplify, we do the exposition in 1D:

$$h_t + (hv)_x = 0$$

$$(hv)_t + (hv^2 + \frac{gh^2}{2})_x = -ghz_x$$

Shock capturing schemes

- Use notation:

$$u = \begin{bmatrix} h \\ hv \end{bmatrix}, f(u) = \begin{bmatrix} hv \\ hv^2 + \frac{gh^2}{2} \end{bmatrix}, s(x,u) = \begin{bmatrix} 0 \\ -ghz_x \end{bmatrix}$$

  so that SWE system can be written as: $\boxed{u_t + f(u)_x = s(x,u)}$.

- Nonlinear hyperbolic system $\Rightarrow$ solutions can develop discontinuities. $\Rightarrow$ use **shock capturing** schemes:

$$u_i^{n+1} = u_i^n - \Delta t \Big(\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} - s_i^n \Big),$$

  where $s_i^n(u(x,t)) \approx s(x_i, u(x_i, t_n))$ and the numerical fluxes $\hat{f}_{i+1/2} = \hat{f}(u_{i-s}, \dots, u_{i+s+1})$ verify

$$\left[\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} \right] (u(x,t)) \approx f(u)_x(x_i, t_n), \quad x_i = i\Delta x, t_n = n\Delta t$$

  and appropriate stability conditions (through **upwinding** and adding numerical viscosity to comply with entropy conditions).

Shock capturing schemes

- Use notation:

$$u = \begin{bmatrix} h \\ hv \end{bmatrix}, f(u) = \begin{bmatrix} hv \\ hv^2 + \frac{gh^2}{2} \end{bmatrix}, s(x,u) = \begin{bmatrix} 0 \\ -ghz_x \end{bmatrix}$$

  so that SWE system can be written as: $\boxed{u_t + f(u)_x = s(x,u)}$.

- Nonlinear hyperbolic system $\Rightarrow$ solutions can develop discontinuities. $\Rightarrow$ use **shock capturing** schemes:

$$u_i^{n+1} = u_i^n - \Delta t \left(\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} - s_i^n \right),$$

  where $s_i^n(u(x,t)) \approx s(x_i, u(x_i, t_n))$ and the numerical fluxes $\hat{f}_{i+1/2} = \hat{f}(u_{i-s}, \ldots, u_{i+s+1})$ verify

$$\left[\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} \right] (u(x,t)) \approx f(u)_x(x_i, t_n), \quad x_i = i\Delta x, t_n = n\Delta t$$

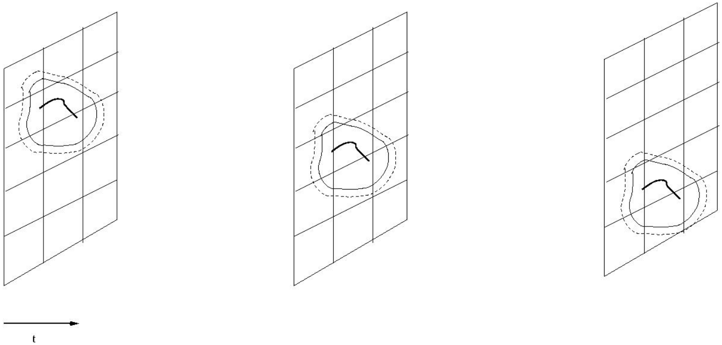
  and appropriate stability conditions (through **upwinding** and adding numerical viscosity to comply with entropy conditions).

Outline

Adaptive schemes

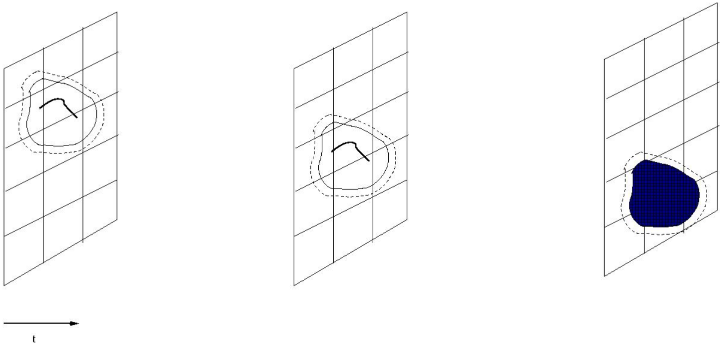
- For $N = 1/\Delta$ and d dimensions, computational cost is $\mathcal{O}(N^{d+1})$, storage is $\mathcal{O}(N^d)$, huge to get small errors.
- Numerical errors are not uniformly distributed:
  - larger errors at discontinuities
  - smaller errors at smooth regions
- An **Adaptive Scheme**, with a smaller Δ where higher errors, would be necessary for $d \geq 2$ and high precision needs.
- Many approaches, we briefly review the (**S**tructured) **A**daptive **M**esh **R**efinement algorithm, proposed by [Berger and Oliger, 1984] and extended by many authors (Colella, Quirk, $\cdots$) to FV schemes.

AMR algorithm



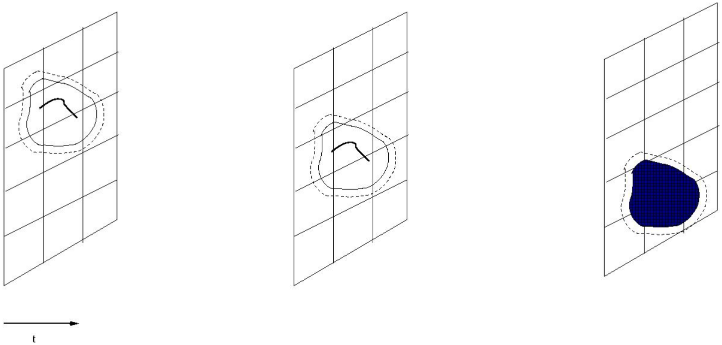
- Time evolution for some grid size $\Delta \equiv \Delta x$ and Δt.

AMR algorithm



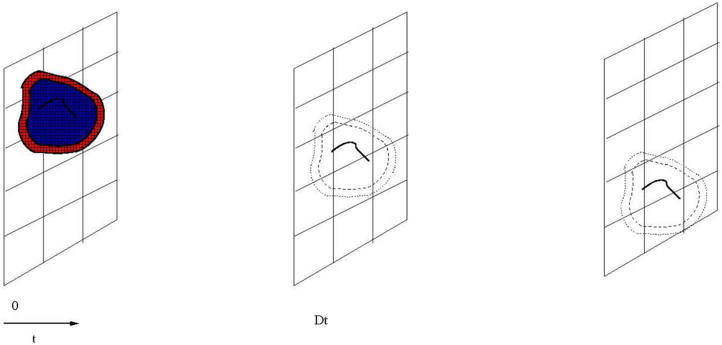
- Want to zoom at **R**egion **O**f **I**nterest, say by using $\Delta/2$.

AMR algorithm



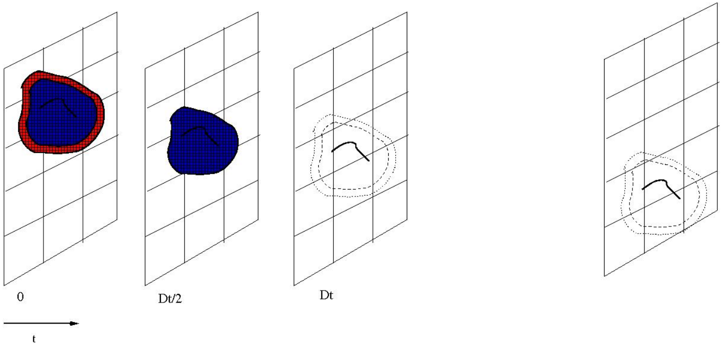
- A: use **interpolation** (zoom), but this causes large errors near shocks.
- B: discard results with Δ, start over with $\Delta/2$.
- C: track region of interest through time evolution.

AMR algorithm



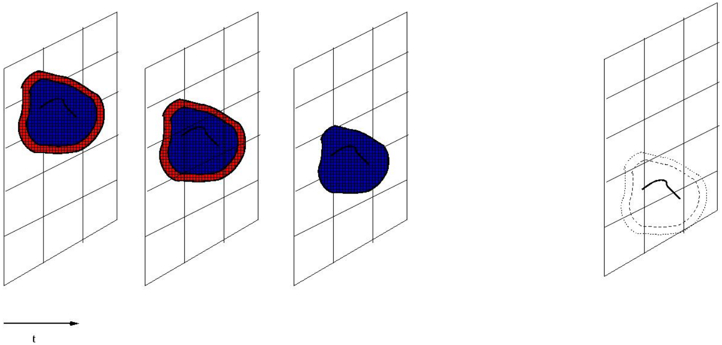
- Before going to B plan, notice that solution on $\Omega \times [0, \Delta t]$ (hopefully) depends on solution at **D**omain of **D**ependence $\widetilde{\Omega} \times \{0\}$ (**by hyperbolicity**).
- Can compute solution at $\Omega \times \{\frac{\Delta t}{2}\}$ (assuming $\Delta/2$ at ROI, same CFL)

AMR algorithm



- How can new DD of region of interest be computed?
- **Zooming by** (x, t)**-interpolation**, OK at (supposedly smooth) surrounding band (**coarse** $\rightarrow$ **fine interpolation**)

AMR algorithm



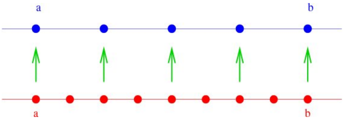
- Recursion $\Rightarrow$ need **nested Grid Hierarchy** (for interpolation).
- Must synchronize data through GH at same (x, t) (**fine $\rightarrow$ coarse project.**)
- More (shorter) time steps at finer resolutions (**local time stepping**).

Outline

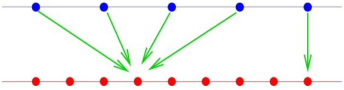
Grid hierarchy

- Grid hierarchy indexed by level l from $l = 0$ (**coarsest**) to $l = L$ (**finest**).
- **Point value approach:** Points in the grid hierarchy: $x_i^l = i\Delta_0/2^l$, $i = 0, \ldots, N_0 2^l$. Since $x_{2i}^{l+1} = x_i^l$ (even indexed points in level $l+1$ are aligned with points in level l), project solution by just copying

$$\mathrm{Proj}_{l+1 \to l}(u^{l+1})_i = u_{2i}^{l+1}, \quad i = 0, \ldots, N_0 2^l.$$



fine $\to$ coarse



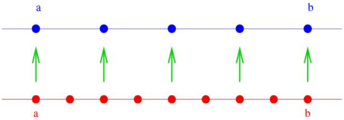
coarse $\to$ fine

- Loss of information when projecting and refining.

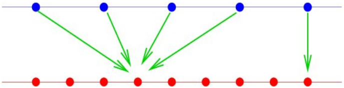
Grid hierarchy

- Grid hierarchy indexed by level l from $l = 0$ (**coarsest**) to $l = L$ (**finest**).
- **Point value approach:** Points in the grid hierarchy: $x_i^l = i\Delta_0/2^l$, $i = 0, \ldots, N_0 2^l$. Since $x_{2i}^{l+1} = x_i^l$ (even indexed points in level $l+1$ are aligned with points in level l), project solution by just copying

$$\mathrm{Proj}_{l+1\to l}(u^{l+1})_i = u_{2i}^{l+1}, \quad i = 0, \ldots, N_0 2^l.$$



fine $\to$ coarse



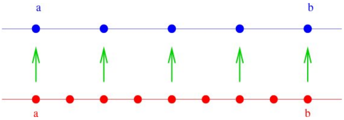
coarse $\to$ fine

- Loss of information when projecting and refining.

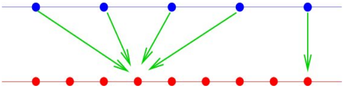
Grid hierarchy

- Grid hierarchy indexed by level l from $l = 0$ (**coarsest**) to $l = L$ (**finest**).
- **Point value approach:** Points in the grid hierarchy: $x_i^l = i\Delta_0/2^l$, $i = 0, \ldots, N_0 2^l$. Since $x_{2i}^{l+1} = x_i^l$ (even indexed points in level $l+1$ are aligned with points in level l), project solution by just copying

$$\mathrm{Proj}_{l+1\to l}(u^{l+1})_i = u_{2i}^{l+1}, \quad i = 0, \ldots, N_0 2^l.$$



fine $\to$ coarse



coarse $\to$ fine

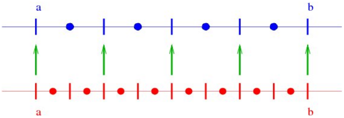
- Loss of information when projecting and refining.

Grid hierarchy

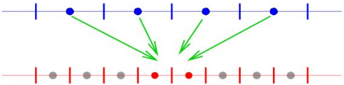
- **Cell-based approach:** Points in the grid hierarchy: $x_i^l = (i + \frac{1}{2})\Delta_0/2^l$, $i = 0, \dots, N_0 2^l - 1$ (**cell centers**).
- Since $\frac{1}{2}(x_{2i}^{l+1} + x_{2i+1}^{l+1}) = x_i^l$, project solution by averaging

$$\mathrm{Proj}_{l+1 \to l}(u^{l+1})_i = \frac{1}{2}(u_{2i}^{l+1} + u_{2i+1}^{l+1}), \quad i = 0, \dots, N_0 2^l - 1.$$

- Cell-interfaces aligned through hierarchy, can copy numerical fluxes $\Rightarrow$ $\boxed{\text{coarse cell average = average of fine cell averages}}$



fine $\to$ coarse



coarse $\to$ fine

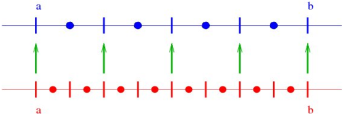
- Can be made conservative.

Grid hierarchy

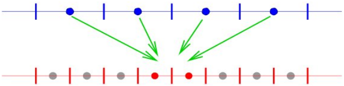
- **Cell-based approach:** Points in the grid hierarchy: $x_i^l = (i + \frac{1}{2})\Delta_0/2^l$, $i = 0, \dots, N_0 2^l - 1$ (**cell centers**).
- Since $\frac{1}{2}(x_{2i}^{l+1} + x_{2i+1}^{l+1}) = x_i^l$, project solution by averaging

$$\mathrm{Proj}_{l+1\to l}(u^{l+1})_i = \frac{1}{2}(u_{2i}^{l+1} + u_{2i+1}^{l+1}), \quad i = 0, \dots, N_0 2^l - 1.$$

- Cell-interfaces aligned through hierarchy, can copy numerical fluxes $\Rightarrow$ $\boxed{\text{coarse cell average = average of fine cell averages}}$



fine $\rightarrow$ coarse



coarse $\rightarrow$ fine

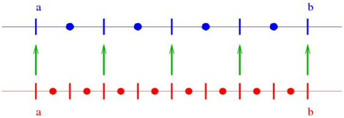
- Can be made conservative.

Grid hierarchy

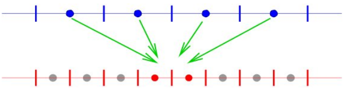
- **Cell-based approach:** Points in the grid hierarchy: $x_i^l = (i + \frac{1}{2})\Delta_0/2^l$, $i = 0, \dots, N_0 2^l - 1$ (**cell centers**).
- Since $\frac{1}{2}(x_{2i}^{l+1} + x_{2i+1}^{l+1}) = x_i^l$, project solution by averaging

$$\mathrm{Proj}_{l+1\to l}(u^{l+1})_i = \frac{1}{2}(u_{2i}^{l+1} + u_{2i+1}^{l+1}), \quad i = 0, \dots, N_0 2^l - 1.$$

- Cell-interfaces aligned through hierarchy, can copy numerical fluxes $\Rightarrow$ coarse cell average = average of fine cell averages



fine $\to$ coarse



coarse $\to$ fine

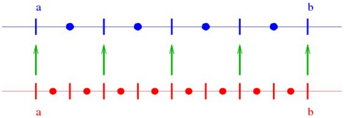
- Can be made conservative.

Grid hierarchy

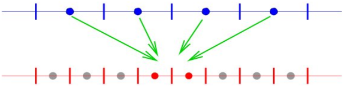
- **Cell-based approach:** Points in the grid hierarchy: $x_i^l = (i + \frac{1}{2})\Delta_0/2^l$, $i = 0, \dots, N_0 2^l - 1$ (**cell centers**).
- Since $\frac{1}{2}(x_{2i}^{l+1} + x_{2i+1}^{l+1}) = x_i^l$, project solution by averaging

$$\mathrm{Proj}_{l+1\to l}(u^{l+1})_i = \frac{1}{2}(u_{2i}^{l+1} + u_{2i+1}^{l+1}), \quad i = 0, \dots, N_0 2^l - 1.$$

- Cell-interfaces aligned through hierarchy, can copy numerical fluxes $\Rightarrow$ $\boxed{\text{coarse cell average = average of fine cell averages}}$



fine $\to$ coarse

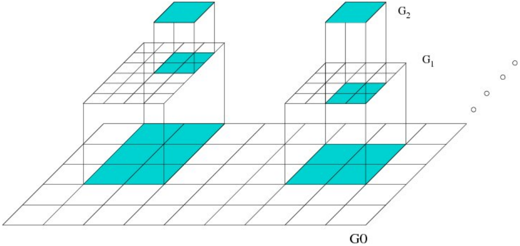


coarse $\to$ fine

- Can be made conservative.

AMR algorithm

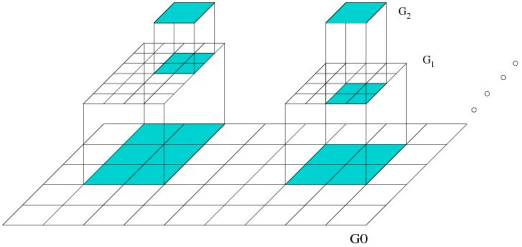
- Nested grids as in 2D example with 2 levels. In a time snapshot we have data where marked. At level 0 all the data is available.



- AMR algorithm ≡ **"time evolution"** of **grid functions** $(u_0^{t_0}, G_0^{t_0}), \dots, (u_L^{t_L}, G_L^{t_L})$ with data $u_l^{t_l}$ attached to grid points indexed by subsets $G_l^{t_l}$ and associated to times $t_0 \geq t_1 \geq \cdots \geq t_L$ (coarser levels evolve "faster" to provide interpolation data to finer levels).
- Index sets $G_l^{t_l}$ have to evolve in time to track flow features (some cells at level l are **marked** by some criterion and are **refined** to give next level $l+1$).

AMR algorithm

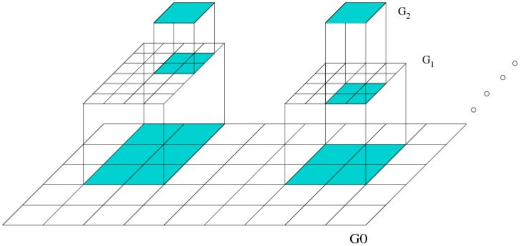
- Nested grids as in 2D example with 2 levels. In a time snapshot we have data where marked. At level 0 all the data is available.



- AMR algorithm $\equiv$ **"time evolution"** of **grid functions** $(u_0^{t_0}, G_0^{t_0}), \dots, (u_L^{t_L}, G_L^{t_L})$ with data $u_l^{t_l}$ attached to grid points indexed by subsets $G_l^{t_l}$ and associated to times $t_0 \geq t_1 \geq \cdots \geq t_L$ (coarser levels evolve "faster" to provide interpolation data to finer levels).
- Index sets $G_l^{t_l}$ have to evolve in time to track flow features (some cells at level l are **marked** by some criterion and are **refined** to give next level $l+1$).

AMR algorithm

- Nested grids as in 2D example with 2 levels. In a time snapshot we have data where marked. At level 0 all the data is available.



- AMR algorithm $\equiv$ **"time evolution"** of **grid functions** $(u_0^{t_0}, G_0^{t_0}), \ldots, (u_L^{t_L}, G_L^{t_L})$ with data $u_l^{t_l}$ attached to grid points indexed by subsets $G_l^{t_l}$ and associated to times $t_0 \geq t_1 \geq \cdots \geq t_L$ (coarser levels evolve "faster" to provide interpolation data to finer levels).
- Index sets $G_l^{t_l}$ have to evolve in time to track flow features (some cells at level l are **marked** by some criterion and are **refined** to give next level $l + 1$).

Outline

Well-balanced schemes

- The convergence of the scheme is usually proved (when possible) through its consistence and stability (this being the harder part).
- But, when seeking convergence to a steady state, it is plausible to require the scheme to **preserve steady states**.
- When the scheme

$$u_i^{n+1} = u_i^n - \Delta t \left(\frac{\hat{f}_{i+1/2}^n - \hat{f}_{i-1/2}^n}{\Delta x} - s_i^n \right)$$

does so, that is:

$$f(u(x))_x = s(x, u(x)) \Longrightarrow \left[\frac{\hat{f}_{i+1/2}^n - \hat{f}_{i-1/2}^n}{\Delta x} - s_i^n \right] (u(x)) = 0$$

then the scheme is termed **well-balanced** [Greenberg and Leroux, 1996].

Well-balanced schemes

- The convergence of the scheme is usually proved (when possible) through its consistence and stability (this being the harder part).
- But, when seeking convergence to a steady state, it is plausible to require the scheme to **preserve steady states**.
- When the scheme

$$u_i^{n+1} = u_i^n - \Delta t\Big(\frac{\hat{f}_{i+1/2}^n - \hat{f}_{i-1/2}^n}{\Delta x} - s_i^n\Big)$$

does so, that is:

$$f(u(x))_x = s(x, u(x)) \Longrightarrow \left[\frac{\hat{f}_{i+1/2}^n - \hat{f}_{i-1/2}^n}{\Delta x} - s_i^n\right](u(x)) = 0$$

then the scheme is termed **well-balanced** [Greenberg and Leroux, 1996].

Well-balanced schemes

- The convergence of the scheme is usually proved (when possible) through its consistence and stability (this being the harder part).
- But, when seeking convergence to a steady state, it is plausible to require the scheme to **preserve steady states**.
- When the scheme

$$u_i^{n+1} = u_i^n - \Delta t\left(\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} - s_i^n\right)$$

does so, that is:

$$f(u(x))_x = s(x, u(x)) \Longrightarrow \left[\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} - s_i^n\right](u(x)) = 0$$

then the scheme is termed **well-balanced** [Greenberg and Leroux, 1996].

Well-balanced schemes

- Special steady state for SWE, **water at rest** ($h + z =$constant, $v = 0$).
- If a scheme preserves this steady state solution, then the scheme is said to verify the C-property [Bermudez and Vazquez, 1994].
- It is not easy to obtain well-balanced schemes: for example, the **centered** choice $s_i^n = s(x_i, u_i^n)$ seldom yields a well-balanced scheme, for this would imply that the finite differencing of the fluxes would be exact (what is not to be expected):

$$s(x_i, u(x_i, t_n)) = f(u(x))_x(x_i, t_n) \qquad \equiv u \textbf{ solution of PDE}$$

$$s(x_i, u(x_i, t_n)) = \left[\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x}\right](u(x)) \qquad \equiv \textbf{well-balancing}$$

- Need to synchronize the discretization of $f(u)_x$ and $s(x, u)$, i.e., need to do an **upwind** discretization of $s(x, u)$.

Well-balanced schemes

- Special steady state for SWE, **water at rest** ($h + z =$constant, $v = 0$).
- If a scheme preserves this steady state solution, then the scheme is said to verify the C-property [Bermudez and Vazquez, 1994].
- It is not easy to obtain well-balanced schemes: for example, the **centered** choice $s_i^n = s(x_i, u_i^n)$ seldom yields a well-balanced scheme, for this would imply that the finite differencing of the fluxes would be exact (what is not to be expected):

$$s(x_i, u(x_i, t_n)) = f(u(x))_x(x_i, t_n) \quad \equiv u \textbf{ solution of PDE}$$

$$s(x_i, u(x_i, t_n)) = \left[\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x}\right](u(x)) \quad \equiv \textbf{well-balancing}$$

- Need to synchronize the discretization of $f(u)_x$ and $s(x, u)$, i.e., need to do an **upwind** discretization of $s(x, u)$.

Well-balanced schemes

- Special steady state for SWE, **water at rest** ($h + z$ =constant, $v = 0$).
- If a scheme preserves this steady state solution, then the scheme is said to verify the C-property [Bermudez and Vazquez, 1994].
- It is not easy to obtain well-balanced schemes: for example, the **centered** choice $s_i^n = s(x_i, u_i^n)$ seldom yields a well-balanced scheme, for this would imply that the finite differencing of the fluxes would be exact (what is not to be expected):

$$s(x_i, u(x_i, t_n)) = f(u(x))_x(x_i, t_n) \quad \equiv u \textbf{ solution of PDE}$$

$$s(x_i, u(x_i, t_n)) = \left[\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x}\right](u(x)) \quad \equiv \textbf{well-balancing}$$

- Need to synchronize the discretization of $f(u)_x$ and $s(x, u)$, i.e., need to do an **upwind** discretization of $s(x, u)$.

Well-balanced schemes

- Special steady state for SWE, **water at rest** ($h + z =$constant, $v = 0$).
- If a scheme preserves this steady state solution, then the scheme is said to verify the C-property [Bermudez and Vazquez, 1994].
- It is not easy to obtain well-balanced schemes: for example, the **centered** choice $s_i^n = s(x_i, u_i^n)$ seldom yields a well-balanced scheme, for this would imply that the finite differencing of the fluxes would be exact (what is not to be expected):

$$s(x_i, u(x_i, t_n)) = f(u(x))_x(x_i, t_n) \quad \equiv u \textbf{ solution of PDE}$$

$$s(x_i, u(x_i, t_n)) = \left[\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x}\right](u(x)) \quad \equiv \textbf{well-balancing}$$

- Need to synchronize the discretization of $f(u)_x$ and $s(x, u)$, i.e., need to do an **upwind** discretization of $s(x, u)$.

Outline

Homogeneous discretization

- We build on [Gascón and Corberán, 2001, Caselles et al., 2009, Donat and Martínez-Gavara, 2011]: we can re-write PDE in "homogeneous" form:

$$u_t + f(u)_x = s(x, u) \Leftrightarrow u_t + g[u]_x = 0$$

  where the **functional** g (dependent on f and s) acts on $u = u(x, t)$ as:

$$g[u](x, t) = f(u(x, t)) + b[u](x, t), \quad b[u](x, t) = -\int_0^x s(r, u(r, t))\, dr$$

- Notice that $u(x)$ is a (stationary) solution of PDE $\Leftrightarrow g[u]_x = 0$.
- We can derive upwind numerical methods for **non-homogeneous** conservation law from well established techniques for **homogeneous** conservation laws.

Homogeneous discretization

- We build on [Gascón and Corberán, 2001, Caselles et al., 2009, Donat and Martínez-Gavara, 2011]: we can re-write PDE in “homogeneous” form:

$$u_t + f(u)_x = s(x,u) \Leftrightarrow u_t + g[u]_x = 0$$

where the **functional** g (dependent on f and s) acts on $u = u(x,t)$ as:

$$g[u](x,t) = f(u(x,t)) + b[u](x,t), \quad b[u](x,t) = -\int_0^x s(r,u(r,t))\,dr$$

- Notice that $u(x)$ is a (stationary) solution of PDE $\Leftrightarrow g[u]_x = 0$.
- We can derive upwind numerical methods for **non-homogeneous** conservation law from well established techniques for **homogeneous** conservation laws.

Homogeneous discretization

- We build on [Gascón and Corberán, 2001, Caselles et al., 2009, Donat and Martínez-Gavara, 2011]: we can re-write PDE in “homogeneous” form:

  $$u_t + f(u)_x = s(x, u) \Leftrightarrow u_t + g[u]_x = 0$$

  where the **functional** g (dependent on f and s) acts on $u = u(x, t)$ as:

  $$g[u](x, t) = f(u(x, t)) + b[u](x, t), \quad b[u](x, t) = -\int_0^x s(r, u(r, t))\, dr$$

- Notice that $u(x)$ is a (stationary) solution of PDE $\Leftrightarrow g[u]_x = 0$.
- We can derive upwind numerical methods for **non-homogeneous** conservation law from well established techniques for **homogeneous** conservation laws.

Homogeneous discretization

- [Donat and Martínez-Gavara, 2011] propose a **Lax-Wendroff**-type discretization for $u_t + g[u]_x = 0$, which is hybridized with a first order monotone scheme through **flux-limiting** techniques that can be written in terms of $\Delta\hat{g}^n_{i-\frac{1}{2}}, \Delta\hat{g}^n_{i+\frac{1}{2}}$ as follows:

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x}(\overbrace{A_i^n \Delta\hat{g}^n_{i-\frac{1}{2}} + B_i^n \Delta\hat{g}^n_{i+\frac{1}{2}}}^{G^n_{i+\frac{1}{2}} - G^n_{i-\frac{1}{2}}})$$

where $G_{i+\frac{1}{2}}$ are numerical fluxes for $g[u]$ and:

$$g_i^n = g[u](x_i, t_n) = f(u(x_i, t_n)) - \int_0^{x_i} s(r, u(r, t_n))dr$$

$$\Delta g^n_{i+\frac{1}{2}} = g^n_{i+1} - g^n_i = f(u(x_{i+1}, t_n)) - f(u(x_i, t_n)) - \underbrace{\int_{x_i}^{x_{i+1}} s(r, u(r, t_n))dr}_{b^n_{i,i+1}}$$

$$\Delta g^n_{i+\frac{1}{2}} \approx \Delta\hat{g}^n_{i+\frac{1}{2}} := f(u^n_{i+1}) - f(u^n_i) + \hat{b}^n_{i,i+1}, \text{ for some } \hat{b}^n_{i,i+1} \approx b^n_{i,i+1}.$$

- For SWE, suitable $\hat{b}^n_{i,i+1}$ can be defined to get **exact C-property**.

Homogeneous discretization

- [Donat and Martínez-Gavara, 2011] propose a **Lax-Wendroff**-type discretization for $u_t + g[u]_x = 0$, which is hybridized with a first order monotone scheme through **flux-limiting** techniques that can be written in terms of $\Delta\hat{g}^n_{i-\frac{1}{2}}, \Delta\hat{g}^n_{i+\frac{1}{2}}$ as follows:

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x}(\overbrace{A_i^n \Delta\hat{g}^n_{i-\frac{1}{2}} + B_i^n \Delta\hat{g}^n_{i+\frac{1}{2}}}^{G^n_{i+\frac{1}{2}} - G^n_{i-\frac{1}{2}}})$$

where $G_{i+\frac{1}{2}}$ are numerical fluxes for $g[u]$ and:

$$g_i^n = g[u](x_i, t_n) = f(u(x_i, t_n)) - \int_0^{x_i} s(r, u(r, t_n)dr$$

$$\Delta g^n_{i+\frac{1}{2}} = g^n_{i+1} - g^n_i = f(u(x_{i+1}, t_n)) - f(u(x_i, t_n)) - \underbrace{\int_{x_i}^{x_{i+1}} s(r, u(r, t_n)dr}_{b^n_{i,i+1}}$$

$$\Delta g^n_{i+\frac{1}{2}} \approx \Delta\hat{g}^n_{i+\frac{1}{2}} := f(u^n_{i+1}) - f(u^n_i) + \hat{b}^n_{i,i+1}, \text{ for some } \hat{b}^n_{i,i+1} \approx b^n_{i,i+1}.$$

- For SWE, suitable $\hat{b}^n_{i,i+1}$ can be defined to get **exact C-property**.

Outline

Well-balanced AMR

- Ingredients of AMR algorithm:
  - Basic numerical scheme.
  - Coarse to fine communication (interpolation).
  - Fine to coarse communication (projection).
- If AMR algorithm is to preserve stationary solutions $\Rightarrow$ each ingredient should preserve them $\Rightarrow$ need well-balanced interpolation and projection.

Well-balanced AMR

- Ingredients of AMR algorithm:
  - Basic numerical scheme.
  - Coarse to fine communication (interpolation).
  - Fine to coarse communication (projection).
- If AMR algorithm is to preserve stationary solutions $\Rightarrow$ each ingredient should preserve them $\Rightarrow$ need well-balanced interpolation and projection.

Well-balanced interpolation: cell approach

- In cell-based grid hierarchy, projection is given by $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$, where indexes indicate the point the data is attached to.
- If $h_i = h(x_i)$ correspond to a water at rest solution, does $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$ correspond to point values (at $x_{i+\frac{1}{2}}$) of the solution?
- If it were so, from $h(x) = \eta - z(x)$ we get

$$h_{i+\frac{1}{2}} = \frac{1}{2}\Big(h(x_i) + h(x_{i+1})\Big) = \eta - \frac{1}{2}\Big(z(x_i) + z(x_{i+1})\Big)$$
$$h_{i+\frac{1}{2}} = h(x_{i+\frac{1}{2}}) = \eta - z(x_{i+\frac{1}{2}}),$$

so z should verify

$$\frac{z(x_i) + z(x_{i+1})}{2} = z\Big(\frac{x_i + x_{i+1}}{2}\Big), \forall i,$$

which does not hold for general z.

- Projection OK if h_i are cell-averages of stationary solution, but then underlying scheme should preserve them (which is not clear, because the well-balancing of an scheme tipically depends on point values).

Well-balanced interpolation: cell approach

- In cell-based grid hierarchy, projection is given by $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$, where indexes indicate the point the data is attached to.
- If $h_i = h(x_i)$ correspond to a water at rest solution, does $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$ correspond to point values (at $x_{i+\frac{1}{2}}$) of the solution?
- If it were so, from $h(x) = \eta - z(x)$ we get

$$h_{i+\frac{1}{2}} = \frac{1}{2}\Big(h(x_i) + h(x_{i+1})\Big) = \eta - \frac{1}{2}\Big(z(x_i) + z(x_{i+1})\Big)$$
$$h_{i+\frac{1}{2}} = h(x_{i+\frac{1}{2}}) = \eta - z(x_{i+\frac{1}{2}}),$$

  so z should verify

$$\frac{z(x_i) + z(x_{i+1})}{2} = z\Big(\frac{x_i + x_{i+1}}{2}\Big), \forall i,$$

  which does not hold for general z.
- Projection OK if h_i are cell-averages of stationary solution, but then underlying scheme should preserve them (which is not clear, because the well-balancing of an scheme tipically depends on point values).

Well-balanced interpolation: cell approach

- In cell-based grid hierarchy, projection is given by $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$, where indexes indicate the point the data is attached to.
- If $h_i = h(x_i)$ correspond to a water at rest solution, does $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$ correspond to point values (at $x_{i+\frac{1}{2}}$) of the solution?
- If it were so, from $h(x) = \eta - z(x)$ we get

$$h_{i+\frac{1}{2}} = \frac{1}{2}\Big(h(x_i) + h(x_{i+1})\Big) = \eta - \frac{1}{2}\Big(z(x_i) + z(x_{i+1})\Big)$$
$$h_{i+\frac{1}{2}} = h(x_{i+\frac{1}{2}}) = \eta - z(x_{i+\frac{1}{2}}),$$

  so z should verify

$$\frac{z(x_i) + z(x_{i+1})}{2} = z\Big(\frac{x_i + x_{i+1}}{2}\Big), \forall i,$$

  which does not hold for general z.
- Projection OK if h_i are cell-averages of stationary solution, but then underlying scheme should preserve them (which is not clear, because the well-balancing of an scheme tipically depends on point values).

Well-balanced interpolation: cell approach

- In cell-based grid hierarchy, projection is given by $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$, where indexes indicate the point the data is attached to.
- If $h_i = h(x_i)$ correspond to a water at rest solution, does $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$ correspond to point values (at $x_{i+\frac{1}{2}}$) of the solution?
- If it were so, from $h(x) = \eta - z(x)$ we get

$$h_{i+\frac{1}{2}} = \frac{1}{2}\Big(h(x_i) + h(x_{i+1})\Big) = \eta - \frac{1}{2}\Big(z(x_i) + z(x_{i+1})\Big)$$
$$h_{i+\frac{1}{2}} = h(x_{i+\frac{1}{2}}) = \eta - z(x_{i+\frac{1}{2}}),$$

so z should verify

$$\frac{z(x_i) + z(x_{i+1})}{2} = z\Big(\frac{x_i + x_{i+1}}{2}\Big), \forall i,$$

which does not hold for general z.
- Projection OK if h_i are cell-averages of stationary solution, but then underlying scheme should preserve them (which is not clear, because the well-balancing of an scheme tipically depends on point values).

Well-balanced interpolation: point-value approach

- For point value grid hierarchy, the projection from level $l+1$ to level l is given by copying values with even indexes, corresponding to the same point-values, so this projection is automatically well-balanced.
- How can we get a **well-balanced** interpolation?
- If we can re-write $f(u)_x = s(x,u)$ as $V(x,u)_x = 0$, then

$$u(x) \text{ is solution of PDE} \Leftrightarrow V(x, u(x)) \text{ is constant.}$$

- $V(x,u) \equiv$ **equilibrium variables**
- In case of SWE, the equilibrium variables are:

$$V(x, \begin{bmatrix} h \\ hv \end{bmatrix}) = \begin{bmatrix} \frac{v^2}{2} + g(h + z(x)) \\ hv \end{bmatrix}$$

Well-balanced interpolation: point-value approach

- For point value grid hierarchy, the projection from level $l+1$ to level l is given by copying values with even indexes, corresponding to the same point-values, so this projection is automatically well-balanced.
- How can we get a **well-balanced** interpolation?
- If we can re-write $f(u)_x = s(x,u)$ as $V(x,u)_x = 0$, then

$$u(x) \text{ is solution of PDE} \Leftrightarrow V(x,u(x)) \text{ is constant.}$$

- $V(x,u) \equiv$ **equilibrium variables**
- In case of SWE, the equilibrium variables are:

$$V(x, \begin{bmatrix} h \\ hv \end{bmatrix}) = \begin{bmatrix} \frac{v^2}{2} + g(h+z(x)) \\ hv \end{bmatrix}$$

Well-balanced interpolation: point-value approach

- For point value grid hierarchy, the projection from level $l+1$ to level l is given by copying values with even indexes, corresponding to the same point-values, so this projection is automatically well-balanced.
- How can we get a **well-balanced** interpolation?
- If we can re-write $f(u)_x = s(x,u)$ as $V(x,u)_x = 0$, then

$$u(x) \text{ is solution of PDE} \Leftrightarrow V(x,u(x)) \text{ is constant.}$$

- $V(x,u) \equiv$ **equilibrium variables**
- In case of SWE, the equilibrium variables are:

$$V(x, \begin{bmatrix} h \\ hv \end{bmatrix}) = \begin{bmatrix} \frac{v^2}{2} + g(h+z(x)) \\ hv \end{bmatrix}$$

Well-balanced interpolation: point-value approach

- For point value grid hierarchy, the projection from level $l+1$ to level l is given by copying values with even indexes, corresponding to the same point-values, so this projection is automatically well-balanced.
- How can we get a **well-balanced** interpolation?
- If we can re-write $f(u)_x = s(x,u)$ as $V(x,u)_x = 0$, then

$$u(x) \text{ is solution of PDE} \Leftrightarrow V(x,u(x)) \text{ is constant.}$$

- $V(x,u) \equiv$ **equilibrium variables**
- In case of SWE, the equilibrium variables are:

$$V(x, \begin{bmatrix} h \\ hv \end{bmatrix}) = \begin{bmatrix} \frac{v^2}{2} + g(h+z(x)) \\ hv \end{bmatrix}$$

Well-balanced interpolation: point-value approach

- For point value grid hierarchy, the projection from level $l+1$ to level l is given by copying values with even indexes, corresponding to the same point-values, so this projection is automatically well-balanced.
- How can we get a **well-balanced** interpolation?
- If we can re-write $f(u)_x = s(x,u)$ as $V(x,u)_x = 0$, then

$$u(x) \text{ is solution of PDE} \Leftrightarrow V(x,u(x)) \text{ is constant.}$$

- $V(x,u) \equiv$ **equilibrium variables**
- In case of SWE, the equilibrium variables are:

$$V(x, \begin{bmatrix} h \\ hv \end{bmatrix}) = \begin{bmatrix} \frac{v^2}{2} + g(h + z(x)) \\ hv \end{bmatrix}$$

Well-balanced interpolation

- Given interpolator $I((w_i); x)$ (i.e., $I((w_i), x_j) = w_j$), if $V(x, \cdot)$ is bijective onto some relevant range then can define a **well-balanced** interpolator by

$$\tilde{I}((u_i); x) = V(x, \cdot)^{-1}(I((V_i); x)), \quad V_i = V(x_i, u_i)$$

(i.e., $\tilde{I}$ interpolates V variables obtained from u and gets back to u variables)

- Since I preserves constants, then $\tilde{I}$ preserves stationary states:

$V(x, u(x)) = K, \forall x \Rightarrow V(x, \tilde{I}((u(x_i)); x)) = K, \forall x \Rightarrow \tilde{I}((u(x_i)); x)$ is a stationary solution

- For SWE, $V(x, \cdot)$ is not injective, but, if we only want to preserve water at rest solutions can take

$$V(x, \begin{bmatrix} \eta \\ \mu \end{bmatrix}) = \begin{bmatrix} \eta + z(x) \\ \mu \end{bmatrix}, \quad V(x, \cdot)^{-1} \begin{bmatrix} \eta \\ \mu \end{bmatrix} = \begin{bmatrix} \eta - z(x) \\ \mu \end{bmatrix}$$

(i.e.,interpolate total heights, then subtract bottom height) and get an interpolation that preserves water at rest solutions.

Well-balanced interpolation

- Given interpolator $I((w_i);x)$ (i.e., $I((w_i),x_j)=w_j$), if $V(x,\cdot)$ is bijective onto some relevant range then can define a **well-balanced** interpolator by

$$\tilde{I}((u_i);x)=V(x,\cdot)^{-1}(I((V_i);x)),\quad V_i=V(x_i,u_i)$$

(i.e., $\tilde{I}$ interpolates V variables obtained from u and gets back to u variables)

- Since I preserves constants, then $\tilde{I}$ preserves stationary states:

$V(x,u(x))=K,\forall x\Rightarrow V(x,\tilde{I}((u(x_i));x))=K,\forall x\Rightarrow\tilde{I}((u(x_i));x)$ is a stationary solution

- For SWE, $V(x,\cdot)$ is not injective, but, if we only want to preserve water at rest solutions can take

$$V(x,\begin{bmatrix}\eta\\ \mu\end{bmatrix})=\begin{bmatrix}\eta+z(x)\\ \mu\end{bmatrix},\quad V(x,\cdot)^{-1}\begin{bmatrix}\eta\\ \mu\end{bmatrix}=\begin{bmatrix}\eta-z(x)\\ \mu\end{bmatrix}$$

(i.e.,interpolate total heights, then subtract bottom height) and get an interpolation that preserves water at rest solutions.

Well-balanced interpolation

- Given interpolator $I((w_i);x)$ (i.e., $I((w_i),x_j)=w_j$), if $V(x,\cdot)$ is bijective onto some relevant range then can define a **well-balanced** interpolator by

$$\tilde{I}((u_i);x)=V(x,\cdot)^{-1}(I((V_i);x)),\quad V_i=V(x_i,u_i)$$

(i.e., $\tilde{I}$ interpolates V variables obtained from u and gets back to u variables)

- Since I preserves constants, then $\tilde{I}$ preserves stationary states:

$V(x,u(x))=K,\forall x\Rightarrow V(x,\tilde{I}((u(x_i));x))=K,\forall x\Rightarrow \tilde{I}((u(x_i));x)$ is a stationary solution

- For SWE, $V(x,\cdot)$ is not injective, but, if we only want to preserve water at rest solutions can take

$$V(x,\begin{bmatrix}\eta\\ \mu\end{bmatrix})=\begin{bmatrix}\eta+z(x)\\ \mu\end{bmatrix},\quad V(x,\cdot)^{-1}\begin{bmatrix}\eta\\ \mu\end{bmatrix}=\begin{bmatrix}\eta-z(x)\\ \mu\end{bmatrix}$$

(i.e., interpolate total heights, then subtract bottom height) and get an interpolation that preserves water at rest solutions.

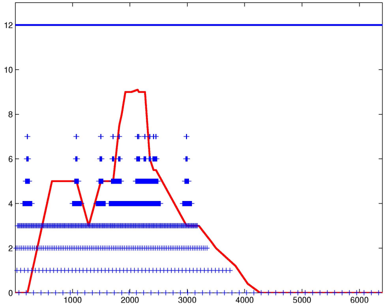
Outline

Tests setup

- We use point-value-based grid hierarchy, with well-balanced interpolation based on linear interpolation.
- Refinement criterion: mark cells to refine when interpolation error exceeds some relative error **rtol** on the maximum interpolation error at each level.
- Δt is adjusted according to the maximum of characteristic speeds to get a Courant number of 0.4.

Test for stationary solutions

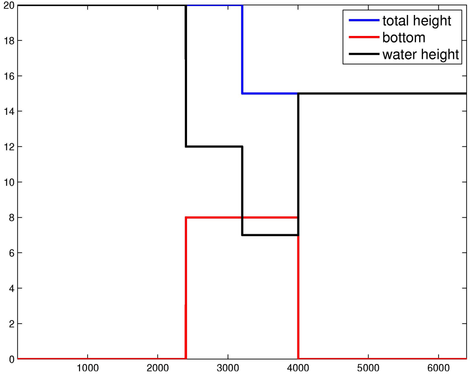
- Water at rest solution of total height=12, bottom topography below. Solution at $T = 200$.
- Have used rtol=10^{-1} $N_0 = 50$, and eight levels ($L = 7$, $N_7 = 6400$) to obtain:



- 4.59% of total integrations (with respect to equivalent finest fixed grid computation), CPU speedup ≈ 11.5.
- Scheme gives approximated solution with water height h such that $||h + z - 12||_\infty = 1.06 \cdot 10^{-14}$ and $||v||_\infty = 3.36 \cdot 10^{-14} \Rightarrow$ **C-property OK to double precision**.

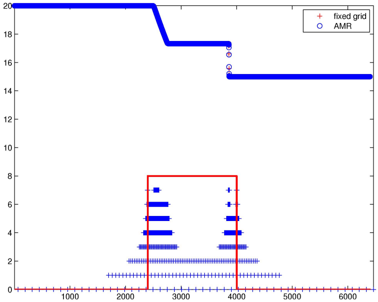
Test for non stationary 1D solutions

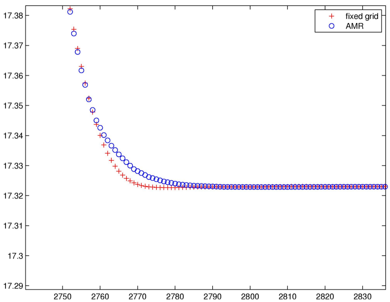
- Dam break problem with bottom topography. Solution at $T = 15$:



Test for non stationary 1D solutions

- Have used rtol=10^{-3}, $N_0 = 50$, and eight levels ($L = 7$, $N_7 = 6400$) to obtain:

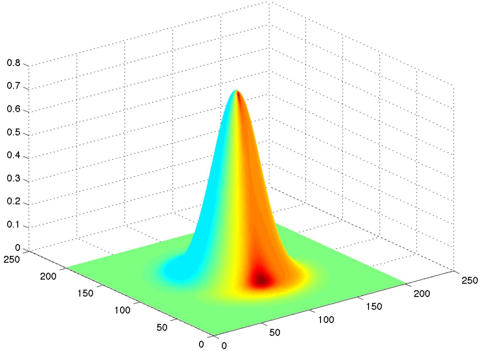




- 4.74% of total integrations (with respect to equivalent finest fixed grid computation), CPU speedup ≈ 14.04.
- Scheme gives approximated solution with water height h such that $||h_{AMR} - h_{fixed}|| = 1.44 \cdot 10^{-4}$, $||v_{AMR} - v_{fixed}|| = 1.47 \cdot 10^{-4}$

Test for stationary 2D solutions

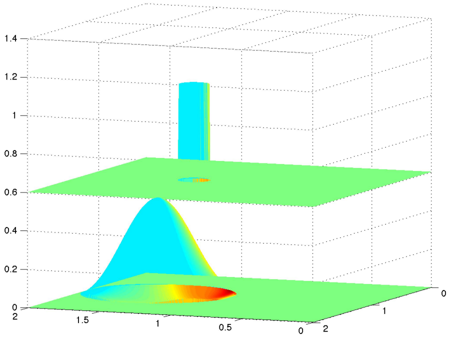
- Water at rest, total height$= 1$ and bottom:



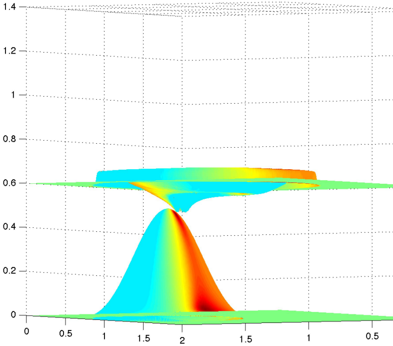
- Have used rtol=10^{-1}, $N_0 = 25$, and 4 levels ($L = 3$, $N_3 = 200$), $T = 0.1$ to obtain: $||h + z - 1||_\infty = 1.1102e - 15$, $||v^x||_\infty = 3.5162e - 15$, $||v^y||_\infty = 3.8820e - 15 \Rightarrow$ **C-property OK to double precision.**
- 22.77% of total integrations, cpu speedup=3.96

Test for non stationary 2D solutions

- Circular dam break problem. Have used rtol=10^{-1}, $N_0 = 25$, and 5 levels ($L = 4$, $N_4 = 400$), $T = 0.25$



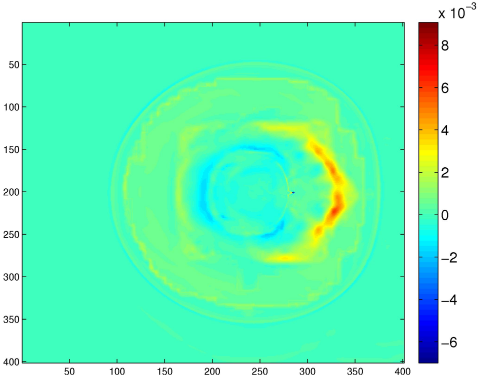
$T = 0$



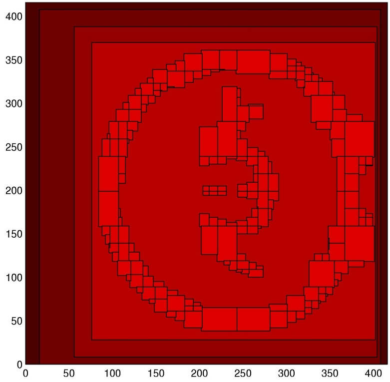
$T = 0.25$

- 29.22% of total integrations, cpu speedup=3.72
- $\|h_{AMR} - h_{fixed}\|_\infty = 0.009$, difference of mass$\approx$ 0.02%.

Test for non stationary 2D solutions



$h_{AMR} - h_{fixed}$



Grid hierarchy (the lighter, the finer)

Conclusions

- We have presented a technique for obtaining well-balanced point-value-based adaptive mesh refinement schemes for shallow water equations.
- We have seen some of the difficulties for getting a well-balanced cell-based adaptive mesh refinement schemes for SWE.
- We have tested the scheme with Donat&Martinez-Gavara homogenized SWE solver and we have obtained an adaptive scheme with the exact C-property.
- We are working on extending it to deal with dry zones.
- Possibility of getting an adaptive scheme that preserves more stationary solutions if underlying scheme does so.

Baeza, A. and Mulet, P. (2006).
Adaptive mesh refinement techniques for high-order shock capturing schemes for multi-dimensional hydrodynamic simulations.
Internat. J. Numer. Methods Fluids, 52(4):455–471.

Berger, M. J. and Oliger, J. (1984).
Adaptive mesh refinement for hyperbolic partial differential equations.
J. Comput. Phys., 53(3):484–512.

Bermudez, A. and Vazquez, M. E. (1994).
Upwind methods for hyperbolic conservation laws with source terms.
Comput. & Fluids, 23(8):1049–1071.

Caselles, V., Donat, R., and Haro, G. (2009).
Flux-gradient and source-term balancing for certain high resolution shock-capturing schemes.
Comput. & Fluids, 38(1):16–36.

Donat, R. and Martínez-Gavara, A. (2011).
A hybrid second order scheme for shallow water flows.
to appear in APNUM.

Gascón, L. and Corberán, J. M. (2001).

Construction of second-order TVD schemes for nonhomogeneous hyperbolic conservation laws.

J. Comput. Phys., 172(1):261–297.

Greenberg, J. M. and Leroux, A. Y. (1996).

A well-balanced scheme for the numerical processing of source terms in hyperbolic equations.

SIAM J. Numer. Anal., 33(1):1–16.